



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/696,828	10/30/2003	Joseph G. Laura	IDF 2506 (4000-14800)	1867
28003	7590	08/09/2007		
SPRINT			EXAMINER	
6391 SPRINT PARKWAY			CHEN, QING	
KSOPHT0101-Z2100				
OVERLAND PARK, KS 66251-2100			ART UNIT	PAPER NUMBER
			2191	
			MAIL DATE	DELIVERY MODE
			08/09/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

mm

Office Action Summary

Application No.

10/696,828

Applicant(s)

LAURA, JOSEPH G.

Examiner

Qing Chen

Art Unit

2191

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 18 July 2007.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-26 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-26 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 18 July 2007 is/are: a) ☒ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
- ☐ Certified copies of the priority documents have been received.
 - ☐ Certified copies of the priority documents have been received in Application No. _____.
 - ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|--|---|
| 1) <input type="checkbox"/> Notice of References Cited (PTO-892) | 4) <input type="checkbox"/> Interview Summary (PTO-413) |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | Paper No(s)/Mail Date. _____ |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO/SB/08) | 5) <input type="checkbox"/> Notice of Informal Patent Application |
| Paper No(s)/Mail Date _____ | 6) <input type="checkbox"/> Other: _____ |

DETAILED ACTION

1. This Office action is in response to the amendment filed on July 18, 2007.
2. **Claims 1-26** are pending.
3. **Claims 1, 6, 11, 19, and 21-24** have been amended.
4. The objection to the drawings is withdrawn in view of Applicant's submission of the correctly labeled replacement drawing sheets.
5. The objection to the specification is withdrawn in view of Applicant's amendments to the specification.
6. The objections to Claims 6, 11, and 19 are withdrawn in view of Applicant's amendments to the claims.
7. The 35 U.S.C. § 112, second paragraph, rejections of Claims 18-20 are withdrawn in view of Applicant's amendments to the claims.
8. The 35 U.S.C. § 101 rejections of Claims 6-20 and 24-26 are withdrawn in view of Applicant's amendments to the claims and the Office's current policies regarding non-statutory subject matter.

Response to Amendment

Claim Objections

9. **Claim 24** is objected to because of the following informalities:
 - **Claim 24** contains a typographical error: "an operating system call" should presumably read -- an operating system --.Appropriate correction is required.

Claim Rejections - 35 USC § 112

10. The following is a quotation of the first paragraph of 35 U.S.C. 112:

The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

11. **Claims 6-26** are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention.

Claims 6-26 recite a computer readable medium as a claimed element. The subject matter is not properly described in the application as filed, since the originally-filed specification lacks disclosure on a computer readable medium. Because the specification does not adequately support the claimed subject matter, it would not reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention.

Claim Rejections - 35 USC § 102

12. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

13. **Claims 1-11, 17-20, and 24-26** are rejected under 35 U.S.C. 102(e) as being anticipated by **Gungabeesoon** (US 7,007,278).

As per **Claim 1**, Gungabeesoon discloses:

- a memory block (*see Figure 1: 102*);
- a COBOL program communicating with the memory block (*see Figure 1: 122*;

Column 11: 23-27, "... it is to be understood that the architecture but could also support legacy applications written in COBOL ...");

- a socket (*see Figure 6: 626A and 626B*); and
- a COBOL routine callable from the COBOL program, the COBOL routine reads

information from the socket and writes the information read from the socket to the memory block in response to the COBOL program call, wherein the COBOL routine reads the

information from the socket through a call to an operating system (*see Figure 5: 410; Figure 6;*

Column 4: 53-58, "Operating system 120 and applications 122 reside in memory 102."; *Column*

10: 62-65, "When the legacy application 122 reaches an I/O instruction, output data is sent as in

Art Unit: 2191

step 632 to the application runtime component 430 of the computer's operating system ...";

Column 11: 13-18, "The input data is then forwarded to socket or queue 626a as in step 642c, to the other application socket or queue 626b and I/O buffers if any and to the application runtime component 430, and eventually to the legacy program 122 that was waiting on a Read_Data method 640b.").

As per **Claim 2**, the rejection of **Claim 1** is incorporated; and Gungabeesoon further discloses:

- wherein the COBOL program further communicates with the COBOL routine to initiate the COBOL routine communication with the socket and the memory block (*see Figure 6; Column 11: 8-10, "Subsequent interactions between the client interface on the network user agent 570 and the application 122 flows through the socket connections 626a and 626b."*).

As per **Claim 3**, the rejection of **Claim 1** is incorporated; and Gungabeesoon further discloses:

- wherein the COBOL routine is further defined as a subroutine of the COBOL program (*see Column 8: 14-17, "Each legacy application 122 has data 422 to be input/output to/from the application runtime operating system 430 according to the program I/O code 410 through the compiler runtime 420."*).

As per **Claim 4**, the rejection of **Claim 1** is incorporated; and Gungabeesoon further discloses:

Art Unit: 2191

- wherein the COBOL routine is further defined as a COBOL library having a plurality of routines callable by the COBOL program (*see Figure 6; Column 10: 37-67 through Column 11: 1-22*).

As per **Claim 5**, the rejection of **Claim 1** is incorporated; and Gungabeesoon further discloses:

- wherein the COBOL routine is further defined as a compiler enabled function usable by the COBOL program (*see Column 8: 14-17, "Each legacy application 122 has data 422 to be input/output to/from the application runtime operating system 430 according to the program I/O code 410 through the compiler runtime 420."*).

As per **Claim 6**, Gungabeesoon discloses:

- requesting, by a COBOL program stored on a computer readable medium, information from a socket (*see Column 11: 13-18, "The input data is then forwarded to socket or queue 626a as in step 642c, to the other application socket or queue 626b and I/O buffers if any and to the application runtime component 430, and eventually to the legacy program 122 that was waiting on a Read_Data method 640b."*);

- retrieving, by a COBOL routine stored on a computer readable medium, information from the socket through a call to an operating system (*see Column 11: 13-18, "The input data is then forwarded to socket or queue 626a as in step 642c, to the other application socket or queue 626b and I/O buffers if any and to the application runtime component 430, and eventually to the legacy program 122 that was waiting on a Read_Data method 640b."*);

- writing, by the COBOL routine, information read from the socket to a memory block *(see Column 4: 53-58, "Memory 102 is a random-access semiconductor memory for storing data and programs ... Operating system 120 and applications 122 reside in memory 102.");* and
- reading from the memory block, by the COBOL program, the information *(see Column 4: 53-58, "Memory 102 is a random-access semiconductor memory for storing data and programs ... Operating system 120 and applications 122 reside in memory 102.");*

As per **Claim 7**, the rejection of **Claim 6** is incorporated; and Gungabeesoon further discloses:

- managing, by the COBOL routine, a connection with the socket *(see Column 10: 51-54, "... the application invoker 660 which creates a socket 626b and makes a connection to the socket in the network server process as in step 620.");*

As per **Claim 8**, the rejection of **Claim 7** is incorporated; and Gungabeesoon further discloses:

- wherein managing includes listening on the socket connection *(see Column 10: 54-57, "After a connection is established in step 622 between the two sockets 626a and 626b, the socket 626b in the network server process waits for data from the legacy program.");*

As per **Claim 9**, the rejection of **Claim 7** is incorporated; and Gungabeesoon further discloses:

Art Unit: 2191

- wherein managing includes disconnecting the connection with the socket (*see Column 11: 20-22, "When the application ends, control returns to the application invoker 660 which closes the endpoint connection as in step 620."*).

As per **Claim 10**, the rejection of **Claim 6** is incorporated; and Gungabeesoon further discloses:

- establishing, by the COBOL routine, a connection with the socket (*see Column 10: 51-54, "... the application invoker 660 which creates a socket 626b and makes a connection to the socket in the network server process as in step 620."*).

As per **Claim 11**, the rejection of **Claim 10** is incorporated; and Gungabeesoon further discloses:

- wherein the connection with the socket is established in response to a request from the COBOL program (*see Column 10: 45-50, "The servlet instance 610, upon receiving the XML or HTTP request for invocation, at step 616 creates a socket 626a ..."*).

As per **Claim 17**, the rejection of **Claim 6** is incorporated; and Gungabeesoon further discloses:

- wherein the COBOL routine further includes a coordination module to coordinate such that the COBOL routine only reads when the socket has information and only writes when the socket is not full (*see Column 10: 57-67 through Column 11: 1-7, "The application invoker 660 writes the descriptor of the application-side socket 626b to an environment variable,*

Art Unit: 2191

activates data redirection through an application programming interface and invokes the legacy application 122, as shown in step 624. ").

As per **Claim 18**, the rejection of **Claim 6** is incorporated; and Gungabeesoon further discloses:

- initiating a call to the operating system by the COBOL routine to establish a socket connection (*see Figure 2: 221; Figure 6; Column 10: 8-17, "For outbound data the servlet instance 610 accepts a data buffer from the socket, looks up the record identifier of the data, instantiates, and populates the associated data object, e.g., the JavaBean, and activate the associated JavaServer Page to serve the data to the network user agent for display. For inbound data, the servlet instance 610 packages the data into a data buffer and submits it to the application via the socket and Publish-to-Web runtime component. ").*

As per **Claim 19**, the rejection of **Claim 18** is incorporated; and Gungabeesoon further discloses:

- wherein the call to the operating system is further defined as a bit-level call to the operating system of a mainframe computer system (*see Figure 2: 221; Figure 6; Column 10: 8-17, "For outbound data the servlet instance 610 accepts a data buffer from the socket, looks up the record identifier of the data, instantiates, and populates the associated data object, e.g., the JavaBean, and activate the associated JavaServer Page to serve the data to the network user agent for display. For inbound data, the servlet instance 610 packages the data into a data buffer and submits it to the application via the socket and Publish-to-Web runtime component. ").*

As per **Claim 20**, the rejection of **Claim 19** is incorporated; and Gungabeesoon further discloses:

- wherein the COBOL routine is further defined as written in COBOL programming language (*see Figure 1: 122; Column 11: 23-27, "... it is to be understood that the architecture but could also support legacy applications written in COBOL ..."*).

As per **Claim 24**, Gungabeesoon discloses:

- reading, by a routine stored on a computer readable medium, information from a socket through a call to an operating system (*see Column 11: 13-18, "The input data is then forwarded to socket or queue 626a as in step 642c, to the other application socket or queue 626b and I/O buffers if any and to the application runtime component 430, and eventually to the legacy program 122 that was waiting on a Read_Data method 640b."*);
- writing, by the routine, the information to an area (*see Column 4: 53-58, "Memory 102 is a random-access semiconductor memory for storing data and programs ... Operating system 120 and applications 122 reside in memory 102."*); and
- reading, by a COBOL program stored on a computer readable medium, the information from the area, the COBOL program and the routine operating in the same runtime environment (*see Figure 6; Column 4: 53-58, "Memory 102 is a random-access semiconductor memory for storing data and programs ... Operating system 120 and applications 122 reside in memory 102."*).

Art Unit: 2191

As per **Claim 25**, the rejection of **Claim 24** is incorporated; and Gungabeesoon further discloses:

- wherein the area is a file (*see Column 11: 2-7, "The network page is populated with data from the data object as in step 652 ..."*).

As per **Claim 26**, the rejection of **Claim 24** is incorporated; and Gungabeesoon further discloses:

- wherein the area is a memory area (*see Figure 1: 102*).

Claim Rejections - 35 USC § 103

14. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

15. **Claims 12-14** are rejected under 35 U.S.C. 103(a) as being unpatentable over

Gungabeesoon (US 7,007,278) in view of Vermeire et al. (US 6,931,623).

As per **Claim 12**, the rejection of **Claim 6** is incorporated; however, Gungabeesoon does not disclose:

- wherein the COBOL routine provides an address to the COBOL program, the address identifying a location of the memory block where the information is written.

Vermeire et al. disclose:

- wherein the COBOL routine provides an address to the COBOL program, the address identifying a location of the memory block where the information is written (*see Column 4: 35-44, "... a reference to the binary data contained within the record layout at the time the programming call to read or write data. The reference to the binary data is most likely a memory address (a "pointer") as implemented in most programming languages."*).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Vermeire et al. into the teaching of Gungabeesoon to include wherein the COBOL routine provides an address to the COBOL program, the address identifying a location of the memory block where the information is written. The modification would be obvious because one of ordinary skill in the art would be motivated to locate data in memory.

As per **Claim 13**, the rejection of **Claim 12** is incorporated; however, Gungabeesoon does not disclose:

- mapping, by the COBOL program, the memory block into the COBOL program.

Vermeire et al. disclose:

- mapping, by the COBOL program, the memory block into the COBOL program (*see Column 6: 43-55, "An existing COBOL copybook, an example of which is shown in FIG. 3, or a PL/I record definition in the source code of an existing legacy application are examples of a source record definition."* and *"The source record definition is processed by a lexical analyzer FIG. 2 capable of translating the language-specific representation of a record layout into a*

language-neutral and computer-architecture neutral representation of the data layout ("metadata"). This metadata is stored on a persistent storage medium 35 of FIG. 12 and accessed and managed via the workbench. ").

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Vermeire et al. into the teaching of Gungabeesoon to include mapping, by the COBOL program, the memory block into the COBOL program. The modification would be obvious because one of ordinary skill in the art would be motivated to locate data in memory.

As per **Claim 14**, the rejection of **Claim 13** is incorporated; however, Gungabeesoon does not disclose:

- wherein the mapping is accomplished using a copybook.

Vermeire et al. disclose:

- wherein the mapping is accomplished using a copybook (*see Column 6: 43-55, "An existing COBOL copybook, an example of which is shown in FIG. 3, or a PL/I record definition in the source code of an existing legacy application are examples of a source record definition."*).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Vermeire et al. into the teaching of Gungabeesoon to include wherein the mapping is accomplished using a copybook. The modification would be obvious because one of ordinary skill in the art would be motivated to describe the physical layout of data.

16. **Claims 15 and 16** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Gungabeesoon** (US 7,007,278) in view of **Ahmad et al.** (US 5,745,748).

As per **Claim 15**, the rejection of **Claim 6** is incorporated; however, **Gungabeesoon** does not disclose:

- wherein the information is provided in an EBCDIC format and wherein the method further comprises converting the information from the EBCDIC format to an ASCII format.

Ahmad et al. disclose:

- wherein the information is provided in an EBCDIC format and wherein the method further comprises converting the information from the EBCDIC format to an ASCII format (*see Column 3: 18-21, "... if the data to be downloaded are in the EBCDIC format, as is common for mainframe computers, it must often be converted to the ASCII format for PC storage or use."*).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of **Ahmad et al.** into the teaching of **Gungabeesoon** to include wherein the information is provided in an EBCDIC format and wherein the method further comprises converting the information from the EBCDIC format to an ASCII format. The modification would be obvious because one of ordinary skill in the art would be motivated to store or use the information in a PC (*see Ahmad et al. – Column 3: 18-21*).

As per **Claim 16**, the rejection of **Claim 15** is incorporated; however, **Gungabeesoon** does not disclose:

Art Unit: 2191

- wherein the conversion is accomplished by the COBOL routine.

Ahmad et al. disclose:

- wherein the conversion is accomplished by the COBOL routine (*see Column 3: 52-56, "... a system and method were needed to enable a mainframe-class application program under development in a PC-based COBOL development system to directly access data on a mainframe computer to which the PC was electronically linked."*).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Ahmad et al. into the teaching of Gungabeesoon to include wherein the conversion is accomplished by the COBOL routine. The modification would be obvious because one of ordinary skill in the art would be motivated to perform the conversion to allow access to mainframe computer data (*see Ahmad et al. – Column 3: 18-21*).

17. **Claims 21-23** are rejected under 35 U.S.C. 103(a) as being unpatentable over Gungabeesoon (US 7,007,278).

As per **Claim 21**, Gungabeesoon discloses:

- a memory block (*see Figure 1: 102*);
- a COBOL program stored on a computer readable medium communicating with the memory block (*see Figure 1: 122; Column 11: 23-27, "... it is to be understood that the architecture but could also support legacy applications written in COBOL ..."*); and

Art Unit: 2191

- a COBOL routine stored on a computer readable medium callable from the COBOL program, the COBOL routine reads information and writes the information to the memory block in response to the COBOL program call, wherein the COBOL routine reads the information from the pipe through a call to an operating system (*see Figure 5: 410; Figure 6; Column 4: 53-58, "Operating system 120 and applications 122 reside in memory 102."; Column 10: 62-65, "When the legacy application 122 reaches an I/O instruction, output data is sent as in step 632 to the application runtime component 430 of the computer's operating system ..."; Column 11: 13-18, "The input data is then forwarded to socket or queue 626a as in step 642c, to the other application socket or queue 626b and I/O buffers if any and to the application runtime component 430, and eventually to the legacy program 122 that was waiting on a Read_Data method 640b."*).

However, Gungabeesoon does not disclose:

- a pipe.

Official Notice is taken that it is old and well known within the computing art to utilize a pipe. Pipelines are often implemented in a multitasking operating system, by chaining processing elements (processes, threads, etc.). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to include a pipe. The modification would be obvious because one of ordinary skill in the art would be motivated to facilitate communications within the same computer.

As per **Claim 22**, the rejection of **Claim 21** is incorporated; and Gungabeesoon further discloses:

- wherein the memory block is further defined as a mainframe memory block and wherein the COBOL program and the COBOL routine are operable on a mainframe computer system (see Figure 2: 202; Figure 6; Column 5: 45-46, "FIG. 2 is an example of a network server 200 which may access a legacy application stored on the computer 100.").

As per **Claim 23**, the rejection of **Claim 22** is incorporated; and Gungabeesoon further discloses:

- a create module communicating with a computer system to create a pipe connection (see Figure 6; Column 10: 51-54, "... the application invoker 660 which creates a socket 626b and makes a connection to the socket in the network server process as in step 620.");
- a connect module that promotes attachment to the pipe connection (see Figure 6; Column 10: 51-54, "... the application invoker 660 which creates a socket 626b and makes a connection to the socket in the network server process as in step 620.");
- an open module that opens the pipe connection to promote communication via the pipe connection (see Figure 6; Column 10: 51-54, "... the application invoker 660 which creates a socket 626b and makes a connection to the socket in the network server process as in step 620.");
- a write module that writes information to the pipe connection, the write module verifies that the pipe connection is not full prior to writing information and blocks when the pipe connection is full (see Figure 6; Column 10: 62-66, "When the legacy application 122 reaches an I/O instruction, output data is sent as in step 632 to the application runtime component 430 of

Art Unit: 2191

the computer's operating system which calls the Write_Data method as in 640a to redirect data to the application-side socket 626b. ");

- a read module coupleable to the pipe connection to read information from the pipe connection (*see Figure 6; Column 11: 13-18, "The input data is then forwarded to socket or queue 626a as in step 642c, to the other application socket or queue 626b and I/O buffers if any and to the application runtime component 430, and eventually to the legacy program 122 that was waiting on a Read_Data method 640b. ");*

- a release module to release the pipe connection (*see Figure 6; Column 11: 20-22, "When the application ends, control returns to the application invoker 660 which closes the endpoint connection as in step 620. ");*

- a remove module to remove the pipe connection from the computer system (*see Figure 6; Column 11: 20-22, "When the application ends, control returns to the application invoker 660 which closes the endpoint connection as in step 620. ");* and

- a delete module to delete the pipe connection wherein the pipe connection is closed (*see Figure 6; Column 11: 20-22, "When the application ends, control returns to the application invoker 660 which closes the endpoint connection as in step 620. ").*

Response to Arguments

18. Applicant's arguments filed on July 18, 2007 have been fully considered, but they are not persuasive.

In the remarks, Applicant argues that:

- a) Gungabeesoon does not disclose a COBOL routine callable from the COBOL program that reads information from the socket.

The Office Action relied on disclosure of the operating system runtime 430 in Gungabeesoon to read on the limitations of the COBOL routine. Specifically, the Office Action relied on the disclosure of the Read_Data and Write_Data methods called by the operating system runtime 430 to read data from and write data to the socket 626b, respectively. While these methods may be considered a routine called by the operating system runtime 430, Applicant respectfully submits that these methods are not a COBOL routine as required by the claims. Furthermore, Applicants respectfully submit that these methods are not called by the COBOL program (legacy application 122 as interpreted by the Office Action) as required by the claims. Rather, the Read_Data and Write_Data methods are called by the operating system.

Examiner's response:

- a) Examiner disagrees. Gungabeesoon clearly discloses a COBOL routine callable from the COBOL program, the COBOL routine reads information from the socket (*see Figure 5: 410; Figure 6; Column 10: 62-65, "When the legacy application 122 reaches an I/O instruction, output data is sent as in step 632 to the application runtime component 430 of the computer's operating system ..."; Column 11: 13-18, "The input data is then forwarded to socket or queue 626a as in step 642c, to the other application socket or queue 626b and I/O buffers if any and to the application runtime component 430, and eventually to the legacy program 122 that was waiting on a Read_Data method 640b."*).

Attention is drawn to Figure 5, Element 410, which illustrates the legacy application (COBOL program) containing I/O instructions (COBOL routine), which are executed (callable) by the legacy application. Note that the input data is forwarded to the legacy application via the application socket, which is depicted in Figure 5 and Figure 6, Element 632 (the left right arrow).

In the remarks, Applicant argues that:

b) The combination of the legacy application 122 and the operating system runtime 430 of Gungabeesoon may not reasonably be interpreted as the claimed COBOL program.

In the interview, Examiner Qing Chen presented an interpretation whereby the claimed COBOL program was being read on the combination of the legacy application 122 and the operating system runtime 430. Applicant respectfully submits that a combination of the legacy application 122 and the operating system runtime 430 is not a COBOL program. In an effort to clarify that the claimed COBOL program is not an operating system, Claim 1 has been amended to include the limitation, "wherein the COBOL routine reads the information ... through a ... call to an operating system". As such, it is clear from the claims that the COBOL program, the COBOL routine, and the operating system are all distinct elements. Therefore, as amended herein, Applicant respectfully submits that the combination of the legacy application 122 and the operating system runtime 430 may not reasonably be interpreted as the claimed COBOL program.

Examiner's response:

Art Unit: 2191

- b) Applicant's arguments are moot in view of Examiner's further clarification of Gungabeesoon. See 35 U.S.C. § 102(e) rejection of Claim 1 above.

In the remarks, Applicant argues that:

- c) Gungabeesoon does not disclose the COBOL routine reads the information through a call to an operating system.

As amended, Claim 1 requires that a COBOL program call a COBOL routine which makes a call to an operating system to read data from a socket. Gungabeesoon discloses a legacy application 122 (COBOL program) which operates unaware of any changes in its native environment, an operating system 430 that intercepts I/O data of the legacy application 122, and the operating system 430 calling a Data_Read method to read data from a socket. A search of Gungabeesoon did not result in any disclosure a call to the operating system 430.

Examiner's response:

- c) Examiner disagrees. Gungabeesoon clearly discloses the COBOL routine reads the information from the socket through a call to an operating system (*see Figure 6; Column 11: 13-18, "The input data is then forwarded to socket or queue 626a as in step 642c, to the other application socket or queue 626b and I/O buffers if any and to the application runtime component 430, and eventually to the legacy program 122 that was waiting on a Read_Data method 640b."*).

In the remarks, Applicant argues that:

- d) The Data_Read method of Gungabeesoon is not a COBOL routine as claimed.

As amended herein, Claim t requires that the COBOL routine reads the information from the socket through a call to an operating system. As disclosed in column 10, lines 29-31 of Gungabeesoon, the Read_Data method is called by the operating system 430. in contrast, the claims require the COBOL routine to perform a call to the operating system. Therefore, Applicant respectfully submits that the Read_Data method of Gungabeesoon may not reasonably be interpreted as the claimed COBOL routine.

Examiner's response:

- d) Applicant's arguments are moot in view of Examiner's further clarification of Gungabeesoon. See Examiner's response (a) above.

In the remarks, Applicant argues that:

- e) The legacy application of Gungabeesoon does not control reading data from the socket.

Claim 1 as amended herein requires, "a COBOL routine **callable from the COBOL program**, the COBOL routine reads information from the socket ... **in response to the COBOL program call.**" Therefore, Claim 1 requires that the COBOL program controls the reading of information from the socket. Gungabeesoon discloses, "the legacy application is unaware of any changes in its native environment, thus requiring no code changes to the application" (Column 9, lines 38-40).

Examiner's response:

- e) Examiner has addressed Applicant's arguments in the Examiner's response (a) above.

Note that Applicant did not traverse the Examiner's assertion of Official Notice with regard to Claims 21-23. Therefore, the "old and well known within the computing art" statement is taken to be admitted prior art because Applicant has failed to traverse the Examiner's assertion of Official Notice (see MPEP § 2144.03).

Conclusion

19. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the Examiner should be directed to Qing Chen whose telephone number is 571-270-1071. The

Art Unit: 2191

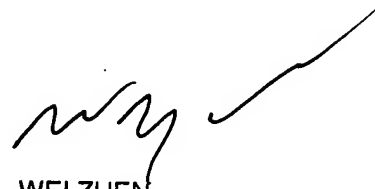
Examiner can normally be reached on Monday through Thursday from 7:30 AM to 4:00 PM.

The Examiner can also be reached on alternate Fridays.

If attempts to reach the Examiner by telephone are unsuccessful, the Examiner's supervisor, Wei Zhen, can be reached on 571-272-3708. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the TC 2100 Group receptionist whose telephone number is 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).



WEI ZHEN
SUPERVISORY PATENT EXAMINER

QC / ac
August 2, 2007